

DOCUMENTAȚIE TEHNICĂ PROIECT

SISTEM ROBOTIC 6-DOF CU FUNCȚIE DE ÎNVĂȚARE ȘI REDARE (TEACH & PLAY)

Autor: Vlad-Mihai BRAȘOVEANU

Data: 14 Ianuarie 2026

CUPRINS

- 1. Introducere și Obiective**
- 2. Arhitectura Sistemului (Hardware)**
- 3. Arhitectura Software și Tehnologii Utilizate**
- 4. Descrierea Funcțională și Moduri de Operare**
- 5. Sisteme de Siguranță și Protecție**
- 6. Provocări Tehnice și Soluții Implementate**
- 7. Direcții Viitoare de Dezvoltare**
- 8. Concluzii**

1. INTRODUCERE ȘI OBIECTIVE

1.1 Prezentare Generală

Acest proiect constă în proiectarea, implementarea și programarea unui braț robotic cu 6 grade de libertate (6-DOF), capabil să manipuleze obiecte și să execute sarcini repetitive. Elementul central al proiectului este funcționalitatea **"Teach & Play"** (Învăță și Redă), care permite utilizatorului să programeze mișcările robotului manual, fără a scrie nicio linie de cod suplimentară, similar roboților industriali de tip "Cobot".

1.2 Obiectivele Proiectului

- Controlul precis al celor 6 servomotoare folosind semnale PWM.
- Implementarea unui sistem de operare în timp real (**FreeRTOS**) pentru gestionarea sarcinilor concurente.
- Crearea unei interfețe om-mașină (HMI) intuitive folosind un ecran LCD și butoane fizice.
- Independența sistemului: robotul funcționează autonom după programare, fără a necesita conexiune permanentă la un PC.

2. ARHITECTURA SISTEMULUI (HARDWARE)

Sistemul este construit modular, având la bază o unitate centrală de procesare care controlează elementele de execuție și citește senzorii.

2.1 Unitatea Centrală: STM32 Nucleo F401RE

Am ales placa de dezvoltare **STM32 Nucleo** datorită performanțelor superioare față de platformele hobby (ex. Arduino):

- **Procesor:** ARM Cortex-M4 la 84 MHz.
- **Periferice:** Timere hardware avansate pentru PWM, ADC rapid pentru senzori, I2C și multiple GPIO-uri.
- **Logica de 3.3V:** Compatibilă cu senzorii moderni.

2.2 Actuatore (Sistemul de Mișcare)

Robotul utilizează 6 servomotoare digitale, mapate astfel:

- **M1 (Bază):** Rotație orizontală (Azimut).
- **M2 (Umăr) & M3 (Cot):** Mișcări principale de elevație.
- **M4 (Ruliu/Roll):** Rotația antebrațului.
- **M5 (Tangaj/Pitch):** Orientarea fină a "mâinii".
- **M6 (Gripper):** Mecanismul de prindere.

Controlul se realizează prin modularea lățimii impulsului (PWM) la 50Hz, generat de Timerele hardware **TIM1** și **TIM3**.

2.3 Interfața cu Utilizatorul și Senzori

- **Panou de Comandă:** 8 Butoane fizice (Select, Plus, Minus, Mode, Record, Play, Home, E-Stop). Butoanele sunt conectate în configurație **Pull-Up Intern** pentru a elimina zgomotul electromagnetic.
- **Afișaj:** LCD 16x2 cu adaptor I2C, pentru afișarea stării curente, unghiurilor și mesajelor de eroare.
- **Senzor de Curent (ACS712):** Monitorizează consumul total al motoarelor pentru a detecta blocaje mecanice sau scurtcircuite.
- **Releu de Putere:** Permite procesorului să taie alimentarea motoarelor în caz de urgență (E-Stop).

3. ARHITECTURA SOFTWARE ȘI TEHNOLOGII

Proiectul software este dezvoltat în **STM32CubeIDE**, folosind limbajul **C**. Arhitectura este bazată pe un sistem de operare în timp real (**RTOS**), esențial pentru a asigura reacția rapidă a robotului.

3.1 FreeRTOS - Sistem de Operare

Aplicația nu rulează într-o buclă infinită simplă (`while(1)`), ci este împărțită în **Task-uri (Sarcini)** independente gestionate de scheduler:

1. **StartTask_Logic (Prioritate Normală):**
 - Este "creierul" robotului.
 - Citește butoanele, interpretează comenzile utilizatorului.

- Gestionează mașina de stări (Manual / Play / Record).
- Calculează interpolarea mișcărilor (funcția `MoveSynchronized`).
- 2. **StartTask_Servo (Prioritate Mare):**
 - Actualizează continuu registrele PWM (CCR) ale timerelor.
 - Convertește unghiurile (grade) în lățime de puls (microsecunde).
- 3. **StartTask_Monitor (Prioritate Critică):**
 - Verifică starea butonului E-STOP și a senzorului de curent la fiecare 20ms.
 - Are autoritatea de a opri sistemul instantaneu.
- 4. **StartTask_HMI (Prioritate Scăzută):**
 - Actualizează ecranul LCD.
 - Rulează asincron pentru a nu încetini mișcările robotului.

3.2 Structuri de Date

Pentru stocarea mișcărilor, am definit o structură personalizată:

```
C
typedef struct {
    float m1, m2, m3, m4, m5, grip;
} RobotPose;
```

Robotul folosește un vector `savedPoints[MAX_POINTS]` pentru a memora traiectoria.

4. DESCRIEREA FUNCȚIONALĂ

Sistemul operează în două moduri principale, comutabile prin butonul MODE.

4.1 Modul MANUAL (Teach)

În acest mod, utilizatorul are control total:

- **Navigare:** Selectarea motorului dorit (M1-M6) cu butonul SELECT.
- **Acționare:** Mișcare fină (+/-) cu butoanele dedicate.
- **Înregistrare:** Butonul RECORD salvează poziția curentă a tuturor axelor în memoria RAM. O apăsare lungă șterge memoria.
- **Funcția HOME:** O apăsare lungă (3 secunde) pe butonul MODE aduce robotul automat în poziția de start (safe position), folosind o interpolare lentă pentru a evita coliziunile.

4.2 Modul PLAY (Automatic)

Odată ce punctele sunt salvate, apăsarea butonului PLAY declanșează execuția autonomă.

- **Interpolare Liniară:** Robotul nu sare brusc între puncte. Software-ul calculează pași intermediari (Smooth Motion) pentru ca toate motoarele să ajungă la destinație simultan.
- **Verificare în Timp Real:** În timpul mișcării automate, sistemul verifică continuu butoanele. Orice apăsare întrerupe imediat ciclul (Stop on Interrupt).
- **Sub-moduri:**
 - *Retrace:* Execută punctele A -> B -> C și se întoarce C -> B -> A.
 - *Fast:* Execută A -> B -> C și sare rapid înapoi la A.

5. SISTEME DE SIGURANȚĂ

Într-un sistem robotic, siguranța este prioritară. Am implementat trei niveluri de protecție.

5.1 E-STOP (Oprire de Urgență)

- **Hardware:** Un buton tip "ciupercă" conectat la pinul PC6.
- **Acțiune:** La apăsare, microcontrolerul dezactivează Releul de putere (tăind alimentarea motoarelor) și blochează execuția software (`isEmergency = 1`).
- **Software:** Funcția de mișcare verifică starea de urgență la fiecare pas al motorului.

5.2 Protecție la Supracurent (Overcurrent)

- Folosind senzorul ACS712 și convertorul ADC al STM32, monitorizăm consumul.
- Dacă curentul depășește pragul de siguranță (calibrat la aprox. 4A) timp de 200ms, sistemul intră în avarie. Aceasta protejează motoarele în cazul în care brațul se blochează mecanic.

5.3 Limite Software (Soft Limits)

- Fiecare motor are limite unghiulare definite în cod (ex: M4 limitat la 80°-170°).
- Aceasta previne coliziunea brațului cu propria structură sau smulgerea cablurilor.

6. DIFICULTĂȚI ÎNTÂMPINATE ȘI SOLUȚII

Procesul de dezvoltare a implicat rezolvarea unor probleme complexe de integrare hardware-software.

6.1 Conflictul de Resurse (Pinout Conflict)

- **Problemă:** Inițial, butonul E-STOP nu funcționa deși era conectat corect.
- **Cauză:** Pinul PB0 era folosit simultan ca intrare pentru buton și ca ieșire PWM pentru Timer 3 (conflict hardware intern).
- **Soluție:** Am identificat conflictul folosind CubeMX și am mutat butoanele critice (Home, E-Stop) pe Portul C (PC10, PC11), care este liber de periferice complexe.

6.2 Intrări Flotante (Floating Inputs)

- **Problemă:** Robotul primea comenzi false sau aleatorii când atingeam firele ("efect de antenă").
- **Cauză:** Rezistențele externe de pull-up aveau contact imperfect pe breadboard.
- **Soluție:** Am activat rezistențele interne de Pull-Up (`GPIO_PULLUP`) ale procesorului STM32. Aceasta a stabilizat intrările logic la 3.3V.

6.3 Blocarea Execuției (Blocking Functions)

- **Problemă:** În timpul mișcării automate, robotul nu răspundea la butonul de Stop.
- **Cauză:** Funcția de mișcare folosea o buclă `for` cu `delay` care bloca procesorul.
- **Soluție:** Am reescris funcția `MoveSynchronized` pentru a verifica starea butoanelor și a variabilei de urgență la fiecare milimetru de deplasare, permițând întreruperea instantanee.

6.4 Interferențe Electromagnetice pe LCD

- **Problemă:** Ecranul afișa caractere eronate ("hieroglife") în timpul funcționării motoarelor.
- **Cauză:** Zgomotul indus de motoare pe liniile I2C lungi.
- **Soluție:** Separarea fizică a cablurilor de semnal față de cele de putere și implementarea unei rutine software de "Refresh" periodic al ecranului.

7. DIRECȚII VIITOARE DE DEZVOLTARE

Deși sistemul este funcțional, există loc pentru îmbunătățiri majore:

1. **Salvarea pe Card SD:** Implementarea unui modul SPI SD Card pentru a salva pozițiile permanent (acum se pierde la resetare).
2. **Cinematica Inversă (IK):** Integrarea bibliotecii `kinematics.c` pentru a permite controlul în coordonate carteziane (X, Y, Z) în locul controlului unghiular direct.
3. **Control prin PC/Bluetooth:** Adăugarea unui modul UART/Bluetooth pentru a controla robotul dintr-o aplicație de telefon sau terminal PC.
4. **Profil de Viteză S-Curve:** Înlocuirea interpolării liniare cu una accelerată/decelerat (S-Curve) pentru mișcări mai fluide și reducerea uzurii mecanice.

8. CONCLUZII

Acest proiect a demonstrat capacitatea de a construi un sistem robotic complex folosind platforma STM32. Utilizarea sistemului de operare FreeRTOS a permis gestionarea eficientă a siguranței și a controlului mișcării simultan.

Robotul rezultat este un instrument educațional puternic, capabil să reproducă sarcini cu o precizie satisfăcătoare, având implementate mecanisme de protecție specifice echipamentelor industriale. Depășirea provocărilor legate de conflictele hardware și zgomotul electric a contribuit semnificativ la robustetea soluției finale.

